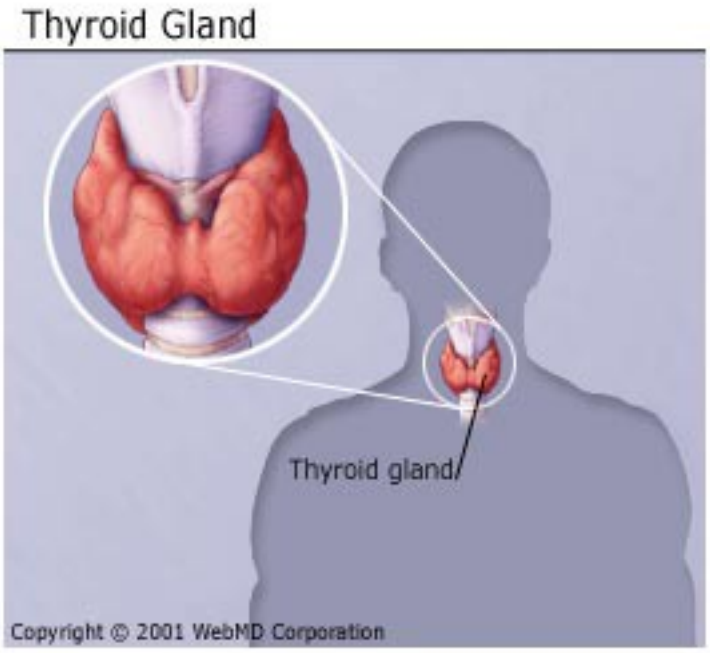
DETAILED PROJECT REPORT

(DPR)

Thyroid disease detection

By HIMANSHU BALODI



https://www.linkedin.com/in/himanshu-balodi-54931124b/

https://github.com/himanshubalodi62/Thyroid_Disease-_Detection_project.git

Contents

Introduction

Importance of Detailed Project Report (DPR)?

1. **Description.**
    1. Problem Perspective.
    2. Problem Statement.
    3. Proposed Solution.
    4. Solution Improvement.
2. **Requirements.**
    1. Hardware Requirements.
    2. Technical Requirements.
3. **Data Requirements.**
    1. Data Collection.
    2. Data Description.
    3. Dataset Characteristic.
    4. License
4. **Data Preprocessing.**
5. **Model Workflow.**
    1. Model Selection
    2. Model Accuracy Score.
6. **Life cycle of a Machine learning project.**
7. **Data Collection / Inputs from the user.**
8. **Data Validation.**
9. **Rendering the results.**
10. **Deployment on Clouds**
11. **Question / Answer**

Conclusion.

Abstract.

This project represents a machine learning-based health Thyroid disease prediction

system. We have used the thyroid dataset from Kaggle, having Features in the dataset that are used for the prediction of the person has thyroid or not include Age, Gender etc. We used Xgboost classifier and determined the relation between binaryClass(which is our dependent feature person has thyroid or not) and these features. We trained the system using an 80-20 split and achieved an accuracy of 99%.

Introduction.

Importance of DPR Documentation?

The main purpose of this DPR documentation is to add the necessary details of the project and provide the description of the machine learning model and written code. This also provides the detailed description on how the entire project has been designed end to end.

Key Points:

- Describes the Design flow
- Implementation
- Software requirements
- Architecture of project
- Non-functional attributes like:
    - Reusability
    - Portability
    - Resource utilization

1. Description.

1. Problem Perspective.

The Thyroid disease Prediction is a hyper-tuned Machine Learning classification model which helps to determine the premium for various policyholders on different parameters.

Parameters such as :- age, sex etc.

2. Problem Statement.

Thyroid disease is one of the most common disease with endocrine disorder in the human population today. For example hyperthyroidism (over) and hypothyroidism (under), which are relate to release of amount of thyroid hormones the thyroid gland produces and whether it is over active trusted source (when thyroid gland makes too much thyroid hormone) or under active trusted source (when the thyroid gland doesn't make enough thyroid hormone). We need to identify whether the patient has thyroid or not.

3. Proposed Solution.

We need to build a ML model which will be used by hospitals and help the hospital authority to identify if the patient has thyroid or not. If it is a positive case then medical will do further test to know what type of thyroid the person is suffering from and according to that the treatment will be on fast-track.

4. Solution Improvements.

The system can be made more futuristic by performing more hyper-tuning methods so that the prediction can be more accurately predictive. The project code has been designed in such a way that whenever new data will come, the model will go under training and if there will be an improvement in the model then the new model will be used for prediction.

2. Requirements

2.1Hardware Requirements:-

- A working computer to code with an active internet connection.

2.2 Tools / Software Requirements:-

- Python version used for this project 3.10 (This may get updated and some features might not be available in new version.)
- Python libraries such as NumPy, Pandas, Matplotlib, Seaborn and scikit-learn (Used for implementation of machine learning algorithms)
- Jupyter Notebook & Visual studio code is used as an IDE for writing the code.
- GitHub is used as the version control system
- AWS is used for deployment using docker image.
- Apache Airflow has been used to monitor the ML model.

3. Data Requirement.

Whenever we are working on any project the data is completely dependent on the requirement of the problem statement. For this project, the problem statement was to create a XGBoost Classifier machine learning model which can predict the thyroid disease person has disease or not based on various parameters.

3.1 Data Collection.

The data which is used in this project was taken from Kaggle.

Dataset link: https://archive.ics.uci.edu/ml/datasets/Thyroid+Disease

3.2 About the dataset.

File Contain.

- Readme.txt

.Thyroid dataset total columns 30 and total raw are 3772 .

3.3 Dataset Characteristic.

Below are the various attributes for the Thyroid Dataset. Please go through the Thyroid_dataset for better clarity of the attributes when reading this document

.age: Age of the patient

.sex: Sex of the patient

Thyroid controls how much energy your body uses (the metabolic rate). It's also involved in digestion, how your heart and muscles work, brain development and bone health. When the thyroid gland does not make enough thyroxine (called hypothyroidism), many of the body's functions slow down.

.on thyroxine: Having thyroxine problem or not?

query on thyroxine: Has query on thyroxine?

on antithyroid medication: Medication is going on or not?

.sick: Sick or not?

.pregnant: Pregnant or not?

.thyroid surgery: Have thyroid surgery or not?

- I131: I-131 is used in medicine to diagnose and treat cancers of the thyroid gland.

.I131 treatment: If having I131 treatment?

.query hypothyroid: Has query on hypothyroid?

query hyperthyroid: Has query on hypothyroid?

- **Lithium:** Lithium may cause hyperthyroidism due to thyroiditis or rarely Graves' disease. As lithium inhibits thyroid hormone release from the thyroid gland it can be used as an adjunct therapy in the management of severe hyperthyroidism.

lithium: Having lithium theraphy?

●**goiter:** A goiter is typically not dangerous, unless the underlying cause of thyroid enlargement is a thyroid cancer.

goiter: Having hoitre or not?
tumor: Having tumor?

● **hypo pituitary:** In hypopituitarism, there is an absence of one or more pituitary hormones. Lack of the hormone leads to loss of function in the gland or organ that it controls.

hypo pituitary: Having hypo pituitary?

●**psych:** The more severe the thyroid disease, the more severe the mood changes(anxiety or depression).

psych: Having psychiatric problems?
TSH measured: Measured?

●**TSH:** TSH stands for thyroid stimulating hormone. A TSH test is a blood test that measures this hormone. TSH levels that are too high or too low may be a sign of a thyroid problem. The thyroid is a small, butterfly-shaped gland in the front of your neck. TSH normal values are 0.5 to 5.0 mIU/L.

T3 measured: Measured?

●**T3**: A T3 test is most often used to diagnose hyperthyroidism, a condition in which the body makes too much thyroid hormone. T3 tests are frequently ordered with T4 and TSH (thyroid stimulating hormone) tests. A T3 test may also be used to monitor treatment for thyroid disease. The range for normal values are: Total T3 – 60 to 180 nanograms per deciliter (ng/dL), or 0.9 to 2.8 nanomoles per liter (nmol/L)

TT4 measured: Measured?

●**TT4:** Thyroxine, also known as T4, is a type of thyroid hormone. A T4 test measures the level of T4 in your blood. Too much or too little T4 can be a sign of thyroid disease. There are two forms of T4 in your blood: Free T4 is the active form of thyroxine hormone that enters your tissues where it's needed. A typical normal range is 0.9 to 2.3 nanograms per deciliter (ng/dL), or 12 to 30 picomoles per liter (pmol/L).

T4U measured: Measured ?

●**T4U:** T4 results that are higher than normal may be a sign of: Hyperthyroidism, which may be caused by Graves disease or another medical condition that causes your thyroid to make too much T4. Thyroiditis (thyroid inflammation) Toxic goiter (an enlarged thyroid with areas that make extra thyroid hormone)

FTI measured: Measured ?

●**FTI:** Free thyroxine index is calculated by dividing the total T4 by the TBI value (T-uptake ratio). Interassay precision (CV) is <5%. Reference range for T-uptake ratio is 0.7–1.2 and for FTI is 6–11.00 µg/dL. The FTI is a normalized determination that remains relatively constant in healthy individuals and compensates for abnormal levels of binding proteins. Hyperthyroidism causes increased FTI, and hypothyroidism causes decreased values.

TBG measured: Measured ?

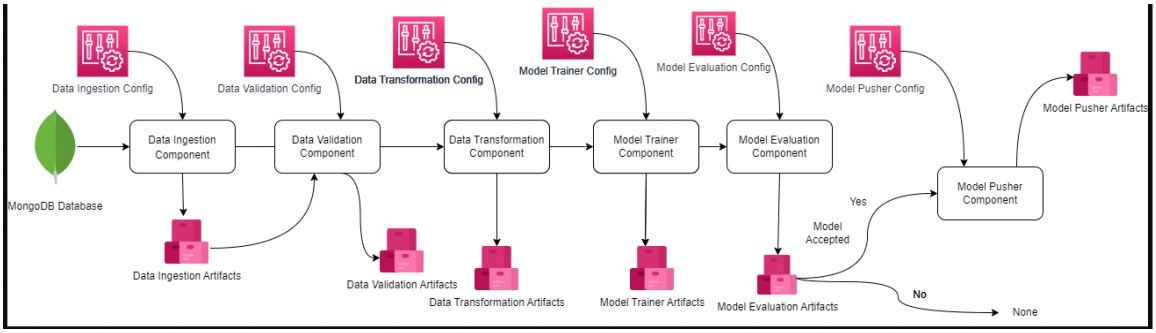
●**TBG:** Thyroxine binding globulin. The TBG blood test measures the level of a protein that moves thyroid hormone throughout your body.

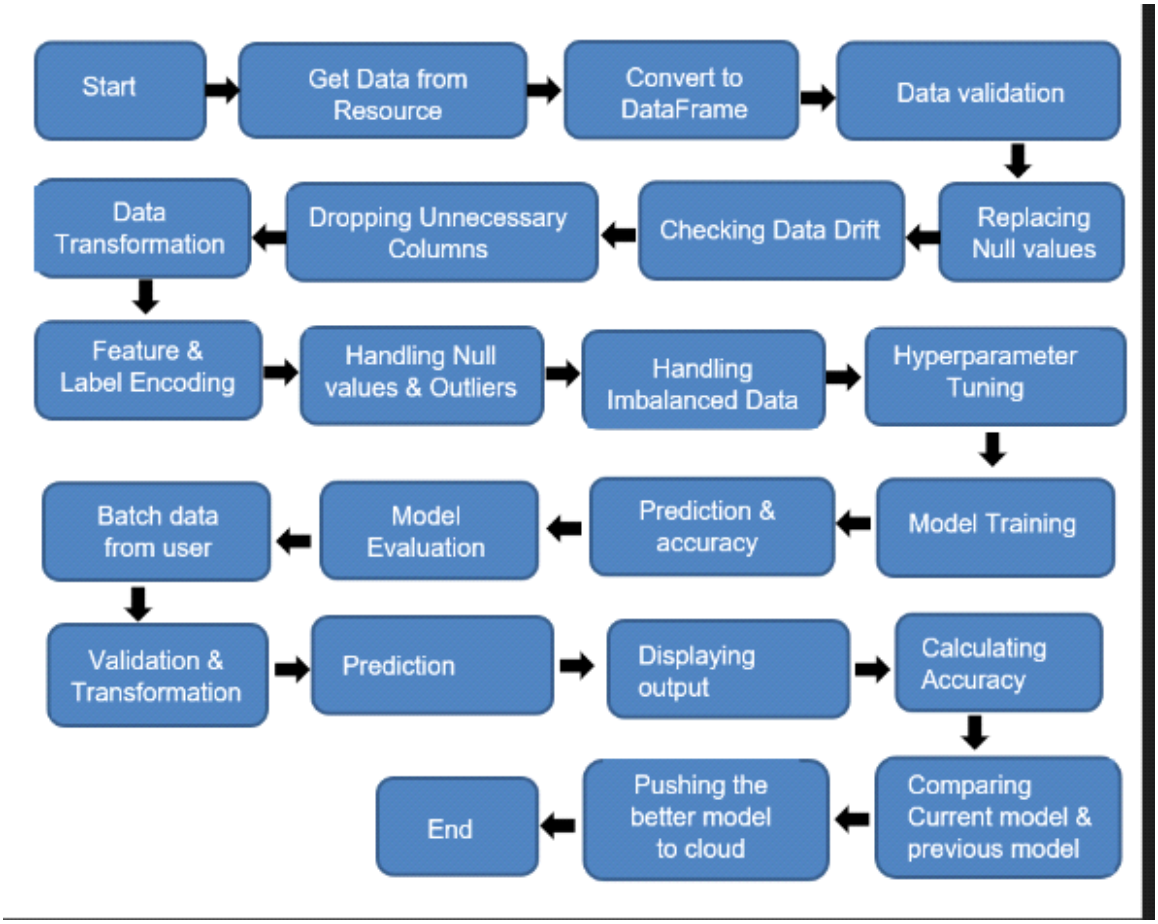
referral source: Source of referral

.binaryClass: Output as P and N

The main part of project is machine learning component part .

Architecture:





.Training pipeline

4. Data Preprocessing.

Have taken the Thyroid.csv file as my dataset.

- All the necessary libraries were imported first such as Numpy, Pandas, Matplotlib, and Seaborn.
- Checking the basic profile of the dataset. To get a better understanding of the dataset.
    - Using Info method
    - Using Describe method

- Checking for unique values of each column.
- Checking for null values, There are no null values present in our dataset.
- Used Matplotlib to plot the basic graphs which is described in the below sections separately

The main part of project is machine learning components part .

Above graph how is going on describe in below.

This is the Initial phase of the project where the given dataset is dumped in the Database(MongoDB) and using that dumped data a notebook file is prepared. At first Data is loaded from the database and after performing all the Exploratory data analysis and Feature engineering we got a good grip over the nature of the data. All the required transformation were also performed. And plotting all the columns how the data distribution Then we started experimenting

There are 6 main components to our project.

1. Data Ingestion

Here we just read the data and we have to create 3 file

(i) training file

(ii) testing file

(iii) validation file

2. Data Validation

Here we are going to validate our data. We have training and testing file and we have our original best (thyroid data) dataset . I want to compare this thyroid file to train file and I wanted to check the both data set column weather they belong same distribution or not . If they are same distribution so ,our train & test file is correct . If not the same so there is data drift so I remove K2-samp means The null hypothesis is that the two identically .

3. Data Transformation

We do standard scaling we are balance to these two classes means which is our target columns(the columns is unbalanced) so we balanced it And our target columns is categorical so we convert it numerical values. Then we compute missing value we apply the robust scale to provide the impact of outliers.

4. Model Trainer

we train our data some eda part to do here & clean the data . We use train our model to machine learning algorithm XGBoost Classifier and also increase the model accuracy we do the hyparameter tuning with Grid-search and load to save our train data in saved model directory with from Trained model.pkl.

5. Model Evaluation

We compare here to our production model with train model If our saved model folder has model then we will compare is best We don’t have model to comparison its happening when we run our training file first time.

Here we are compare testing dataset previous trained object and currently trained model objects.

6. Model Pusher

here we push our model to saved model into model pusher directory .

5. Model Workflow

1.Model Selection

- After this the data was split into 2 sets X and y. X contains all the columns except the target column in our case, y contains only the Target column.
- Using train test split we first split the dataset into X_train,X_test, y_train, y_test .
- The following libraries were imported to create classification with help of KNN imputer models.
    - sklearn.linear_model import , XgBoostClassifier
    - from sklearn.tree import DecisionTreeClassifier,
    - from sklearn.ensemble import RandomForestclassifier, GradientBoostclassifier

2. Model accuracy

Model	Imputation_method	Model_Score
Random Forest	KNN imputer	97.85%
XGBClassifier	KNN imputer	99.85%
Random Forest	Simple Imputer(median)	98.85%
XGBClassifier	Simple Imputer(median)	99.85%
Random Forest	Simple Imputer(Constant)	98.85%
XGBClassifier	Simple Imputer(Constant)	98.85%
Random Forest	Simple Imputer(Mean)	98.85%

Report:

> **Hence XGBoost classifier is giving the better accuracy in overall models. And we will use XGBoost Classifier with KNN imputation for the final mode**l.

6. Life cycle of a Machine learning project

A machine learning project typically goes through the following stages:

1. Problem definition: The first step is to define the problem that you want to solve using machine learning. This involves understanding the business context, determining the goals and objectives of the project, and identifying the data that you will need to train the model.

2. Data collection and preparation: The next step is to collect and prepare the data that you will use to train the model. This may involve scraping data from the web, collecting data from APIs, or using a pr-existing dataset. You will also need to clean and preprocess the data to get it into a suitable format for training.

3. Exploratory data analysis: Once you have collected and prepared the data, you will need to explore it to get a better understanding of its characteristics and any patterns that may exist. This will help you to identify any issues with the data and inform the development of the machine learning model.

4. Preprocessing:
Data preprocessing refers to the process of preparing data for use in training a machine learning model. This typically involves a number of steps, including:

1. **Collecting data from various sources**: This may include scraping data from websites, accessing databases, or collecting data from sensors or other devices.
2. **Cleaning the data:** This involves removing any invalid or missing data, correcting errors, and ensuring that the data is in a consistent format.
3. **Normalizing or scaling the data:** This involves transforming the data so that it is on the same scale, which can help improve the performance of some machine learning algorithms.
4. **Splitting the data into training and test sets:** This involves dividing the data into two sets, one for training the model and the other for evaluating its performance.
5. **Extracting features:** This involves selecting the relevant data and transforming it into a form that can be used by a machine learning algorithm.
6. **Encoding categorical data:** If the data includes categorical variables, they need to be encoded as numerical values in order to be used by most machine learning algorithms.

Data preprocessing is an important step in the machine learning process, as it helps ensure that the data is in a suitable form for training a model and can help improve the model's performance.

4. Model selection and training: The next step is to select an appropriate machine learning model and train it on the data. This will involve selecting a model type, choosing the hyperparameters for the model, and training the model on the data using an optimization algorithm.

5. Model evaluation: Once the model has been trained, you will need to evaluate its performance to see how well it is able to make predictions. This will involve using a variety of evaluation metrics, such as accuracy, precision, and recall, to assess the model's performance.

6. Model deployment: If the model performs well and meets the project goals, it can be deployed to a production environment where it can be used to make predictions in real-time. This may involve integrating the model into a web or mobile application, or using it to automate a business process.

7. Model maintenance: Even after a model has been deployed, it will still need to be maintained and updated over time. This may involve retraining the model on new data, fine-tuning the hyperparameters, or implementing additional features to improve its performance.

Model Retraining

All the models that our system going to use for prediction will be stored in a directory called “saved_models”. Whenever a new or updated data is available in the database, we can initiate our Model Retraining pipeline manually. At the end of this process, new models will get stored in the same directory but in a new folder indicating a new versions of models. As soon as we commit these changes to GitHub, the new model will get deployed automatically.

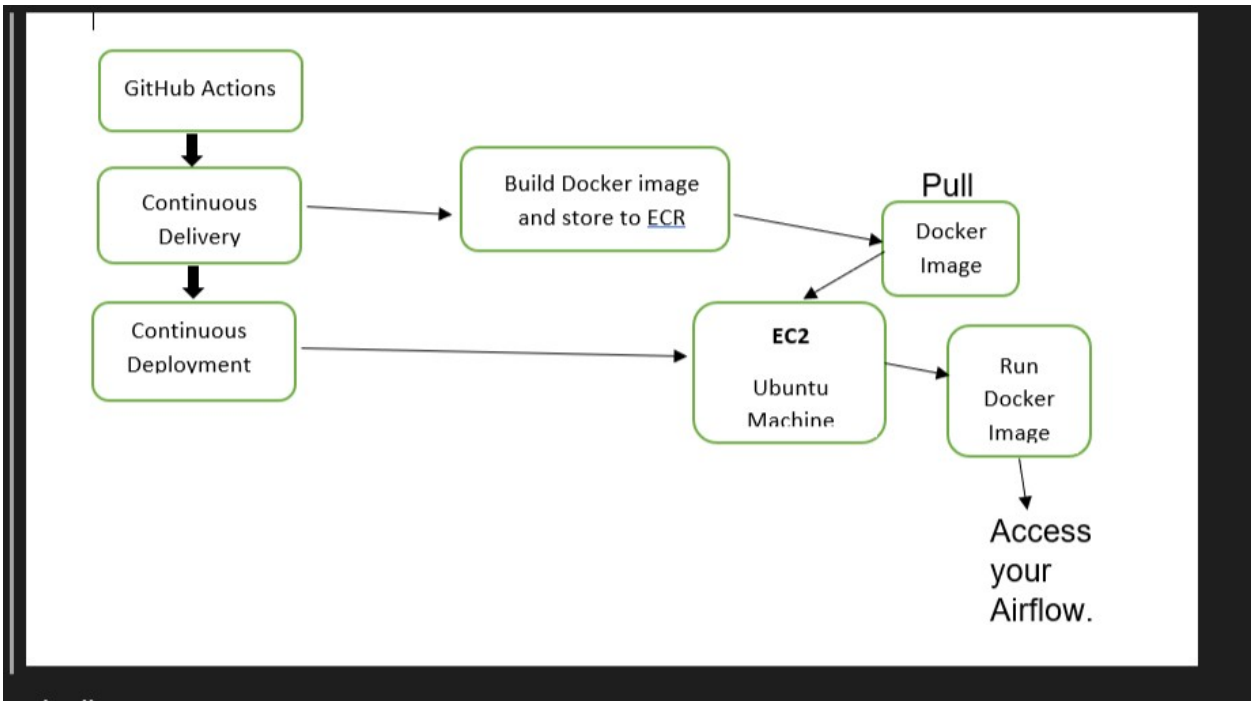
10.Deployment

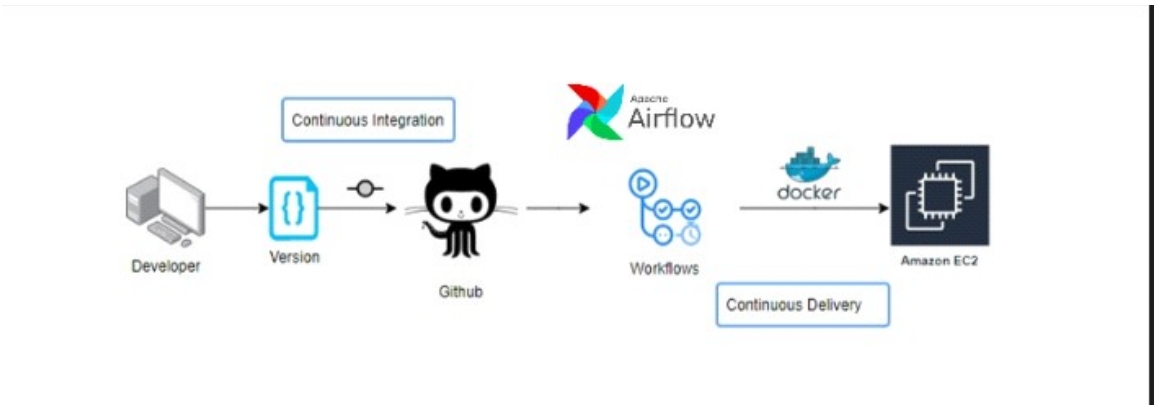
The project is to be delivered using git-Actions workflow and Amazon web Services.

This model is deployed on AWS Ec2 instances. The following are the steps to deploy the model on the AWS platform:

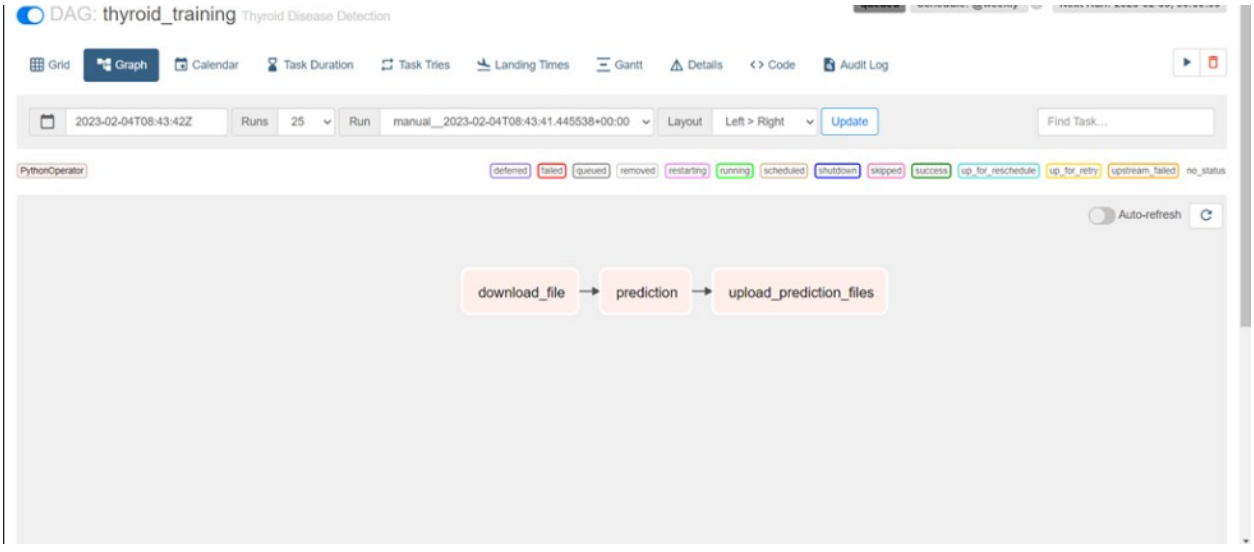
- Create an AWS account
- Create an ECR
- Create S3 bucket

- Create an EC2 instance
- Edit security group
- Connect to an EC2 instance
- Install Docker
- Add the runner in the GitHub
- Add all the secret keys in the GitHub
- In the GitHub actions, run the continuous delivery and deployment workflow once after starting the runner in the ec2 instance
- Start the instance & locate the Docker run.sh file for to initiate the "Runner" to pick the job.
- Use Apache airflow to monitor the model and perform batch prediction





batch_prediction



Question / Answer

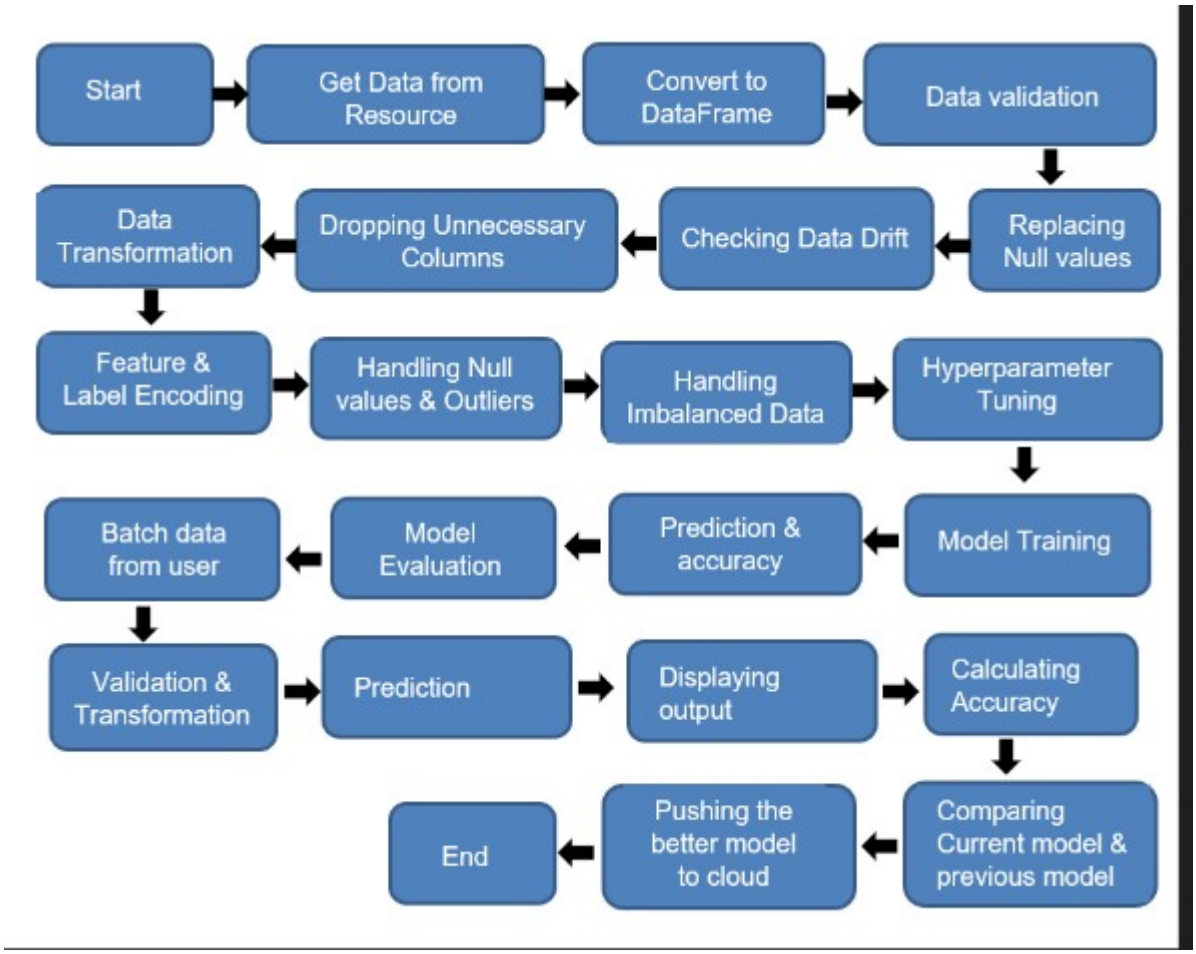
Q1) What's the source of data?

The data for training is provided by the client in multiple batches and each batch contain multiple files

Q 2) What was the type of data?

The data was the combination of numerical and Categorical values.

Q 3) What's the complete flow you followed in this Project?



Q 5) How logs are managed?

We are using different logs as per the steps that we follow in validation and modeling like File validation log , Data Insertion ,Model Training log , prediction log etc.

Q 6) What techniques were you using for data pre-processing?

Removing unwanted attributes

Visualizing relation of independent variables with each other and output variables

Checking and changing Distribution of continuous values

Removing outliers

Cleaning data and imputing if null values are present.

Converting categorical data into numeric values.

Scaling the data

Q 7) How training was done or what models were used?

Before diving the data in training and validation set we performed clustering over fit to divide the data into two part train file and test file. As per these files the training and validation data were divided.

The scaling was performed over training and validation data

Algorithms like XGBoost were used based on the recall final model was used for each cluster and we saved that model .

Q 8) How Prediction was done?

We create a prediction directory then we read input files then loading transformer using this we prepare for input array once we got input array we are loading our model then we are making prediction this prediction is categorical so we convert is numerical values. We load target encoder using this target encoder I will convert this numerical prediction then we are combining everything is existing data frame and prediction and combining categorical prediction and both are combining we write into the saving the prediction file.

Linked-in link :

https://www.linkedin.com/in/himanshu-balodi-54931124b/

GitHub link :

https://github.com/himanshubalodi62/Thyroid_Disease-_Detection_project.git